AMENDMENTS TO THE CLAIMS

Please cancel claims 9-10 without prejudice. Kindly amend claims 1, 2, 4, 5, 6, 7, 8, 11 and 12 as shown in the following listing of claims. The listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims

1. (Currently amended) In a packet processor having a local packet memory (LPM) for storing packet data during processing, the LPM having a plurality of memory cells, the memory cells accessible by at least one memory access port, a system for managing port contention between at least two controllers that access the memory cells, comprising:

a buffer for ~~queueing~~ queuing ~~read~~/write requests to the at least one memory port for a first one of the controllers; and

a logic mechanism associated with the buffer for determining ~~busy status of the port for a pending request to the cell, and for issuing an appropriate command if the port is busy~~whether a write request from said first one of the controllers is within said buffer and is directed at a first one of the memory cells, and if so, whether a read request exists from a second one of the controllers and is directed at said first one of the memory cells;

~~characterized in that the logic mechanism, if the port is determined to be busy, issues a command for a temporary cessation of write requests to the buffer.~~wherein, if a read request exists from said second one of the controllers and is directed at said first one of the memory cells while said write request is buffered, said read request is delayed until said write request has completed regardless of when said read request occurs.

2. (Currently amended) The system of claim 1 wherein said logic mechanism ~~for~~ prevents~~ing~~ download of packet data from the LPM before ~~all~~ queued write requests to the packet data are completed~~, wherein, if the port is busy, the logic mechanism checks the queue fir write requests, and finding a write request, also issues a command to temporarily suspend packet downloads from the LPM until all write requests in the buffer are accomplished~~.

3. (Original) The system of claim 1 wherein the LPM comprises a set of individual memory cells, each cell having an access port.

4. (Currently amended) The system of claim 3 wherein~~-~~if the port is busy, the logic mechanism checks the buffer for write requests, and finding a write request, issues a command to temporarily suspend packet downloads from the LPM until all write requests in said buffer are completed~~the LPM has eight cells, and each cell has two memory access ports~~.

5. (Currently amended) The system of claim 3 wherein packet downloading is managed by a packet management unit (PMU) issuing read requests, and ~~read/~~said write requests sent to ~~the~~ said buffer are other than read requests from ~~the~~ said PMU, ~~the~~ said PMU reads and ~~the~~ said buffer sharing the same port with ~~the~~ said PMU, said PMU reads having priority over reads in said buffer, ~~and~~ wherein, for packets having data extensive enough to occupy two or more ~~lines of a cell~~memory cells, packet data is interleaved among cells, such that consecutive cells are dedicated to different packets, ensuring that port contention is limited to alternate cycles.

6. (Currently amended) The system of claim 3 wherein packets ~~in process~~ are assigned ~~each a specific~~ an identifier, and wherein said write requests in ~~the~~ said buffer are tagged with said ~~packet~~ identifiers, ~~and the~~said logic mechanism ~~issues~~ issuing commands along with ~~the packet id tag~~said identifier.

7. (Currently amended) In a packet processor having a local packet memory (LPM) for storing packet data during processing, the LPM having a plurality of memory cells accessible by at least one access port, a method for managing contention between at least two controllers that access the memory cells~~at a cell port~~, comprising the steps of:

~~(a) queueing~~ queuing ~~read~~/write requests from a first controller to the port in a buffer; and

~~(b)~~ determining, by a logic mechanism associated with the buffer, ~~busy status of the port for a pending request to the cell, and issuing anappropriate command for a temporary cessation of write requests to the buffer if the port is busy~~ whether a write request from the first one of the controllers is within the buffer and is directed at a first one of the memory cells, and if so:

determining whether a read request exists from a second one of the controllers and is directed at the first one of the memory cells; and

if a read request exists from the second one of the controllers and is directed at the first one of the memory cells while the write request is buffered, delaying the read request until said write request has completed, regardless of when said read request occurs.

8. (Currently amended) The method of claim 7 for managing contention between at least two controllers that access memory cells, further comprising:

preventing download of packet data from the LPM before all queued write requests to the data are completed , wherein, in said step ~~(b)~~of determining whether a write request is within the buffer, the logic mechanism checks the buffer for write requests, and finding a write request, ~~also~~ issues a command to temporarily suspend packet downloads from the LPM until ~~all~~ the queued write requests in the buffer are ~~accomplished~~completed.

9-10. (Canceled)

11. (Currently amended) The method of claim [[9]]8 wherein packet downloading is managed by a packet management unit (PMU) issuing read requests, and ~~read/~~write requests sent to the buffer are other than read requests from the PMU~~, the PMU reads and the buffer sharing the same port with the PMU reads having priority, and wherein, for packets having data extensive enough to occupy two or more lines of a cell, packet data is interleaved among cells, such that consecutive cells are dedicated to different packets, ensuring that port contention is limited to alternate cycles~~.

12. (Currently amended) The method of claim [[9]]8 further comprising:

~~wherein packets in process are assigned each a specific~~assigning an identifier to packets in process~~; and wherein write~~ and

tagging requests in the buffer ~~are tagged with packet~~ identifiers; wherein

~~, and~~ the logic mechanism ~~in step (b)~~ issues commands along with the packet ~~id~~ tagidentifiers.